**IN THE UNITED STATES PATENT AND TRADEMARK OFFICE**

| | | |
|---|---|---|
| In re application of: **Arndt et al.** | § | |
| | § | Group Art Unit: **2169** |
| Serial No.: **10/777,724** | § | |
| | § | Examiner: **Black, Linh** |
| Filed: **February 12, 2004** | § | |
| | § | Confirmation No.: **5919** |
| For: **Architecture and Method for Managing the Sharing of Logical Resources Among Separate Partitions of a Logically Partitioned Computer System** | § § | |

35525

PATENT TRADEMARK OFFICE
CUSTOMER NUMBER

**Commissioner for Patents**
**P.O. Box 1450**
**Alexandria, VA 22313-1450**

## APPEAL BRIEF (37 C.F.R. 41.37)

This brief is in furtherance of the Notice of Appeal, filed in this case on December 18, 2008.

An appeal fee for filing an Appeal Brief was paid on December 19, 2007. The Examiner subsequently reopened prosecution, without an appeal being decided. Therefore, no additional fees are believed to be necessary. If, however, any additional fees are required, I authorize the Commissioner to charge these fees which may be required to IBM Corporation Deposit Account No. 09-0447.

## REAL PARTY IN INTEREST

The real party in interest in this appeal is the following party: International Business Machines Corporation of Armonk, New York.

## RELATED APPEALS AND INTERFERENCES

This appeal has no related proceedings or interferences.

# STATUS OF CLAIMS

## A. TOTAL NUMBER OF CLAIMS IN APPLICATION

The claims in the application are: 1-22

## B. STATUS OF ALL THE CLAIMS IN APPLICATION

Claims canceled: 11-21

Claims withdrawn from consideration but not canceled: None

Claims pending: 1-10 and 22

Claims allowed: None

Claims rejected: 1-10 and 22

Claims objected to: None

## C. CLAIMS ON APPEAL

The claims on appeal are: 1-10 and 22

## **STATUS OF AMENDMENTS**

No amendments were submitted after the Final Office Action of October 3, 2008.

## SUMMARY OF CLAIMED SUBJECT MATTER

### A. CLAIM 1 - INDEPENDENT

The subject matter of claim 1 is directed to a method for managing shared resources in a logical partitioned data processing system. The method includes granting, by a server partition in the logical partitioned data processing system, a logical resource owned by the server partition *(Specification, page 15, line 5 – page 17, line 17; Figure 4, 402)* from a server partition in the logical partitioned data processing system *(Specification, page 17, lines 7-17; Figure 4, 410)* to a client partition in the logical partitioned data processing system *(Specification, page 15, lines 7-14; Figure 4, 420)*. The method further includes communicating an identifier *(Specification, page 16, line 23 – page 17, line 6; Figure 5, 504)* from the server partition to the client partition *(Specification, page 17, lines 7-17; Figure 5, 504)*. The method further includes the client partition mapping the logical resource into a logical address space of the client partition *(Specification, page 16, line 23 – page 17, line 17; page 22, lines 8-10; Figure 5, 508, 510)*. The mapping is performed by the client partition responsive to the client partition accepting the identifier *(Specification, page 21, line 24 – page 22, line 11; Figure 5, 508, 510)*.

# GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

The grounds of rejection to review on appeal are as follows:

## A. GROUND OF REJECTION 1

Whether claims 1-10 and 22 fail to be anticipated under 35 U.S.C. § 102 by *Armstrong et al.,* Processor Reset Generated Via Memory Access Interrupt, U.S. Patent No. 6,467,007 (October 15, 2002) (hereinafter "*Armstrong*").

## B. GROUND OF REJECTION 2

Whether claim 22 is non-obvious under 35 U.S.C. § 103 over *Armstrong* in view of *Farber et al.,* Optimized Network Resource Location, U.S. Patent App. No. 2001/0056500 (December 27, 2001) (hereinafter "*Farber*").

# ARGUMENT

## A. GROUND OF REJECTION 1 (Claims 1-10 and 22)

The Examiner rejected claims 1-10 and 22 under 35 U.S.C. §102(b) as being anticipated by *Armstrong*. Appellants request that the Board of Patent Appeals and Interferences overturn this clearly erroneous rejection.

### A.1. Claims 1, 3, 4, 7, 9, and 10

Claim 1 is representative of this claim grouping. Claim 1 is as follows:

> 1. A method for managing shared resources in a logical partitioned data processing system, the method comprising:
>
> granting, by a server partition in the logical partitioned data processing system, a logical resource owned by the server partition to a client partition in the logical partitioned data processing system;
>
> communicating an identifier from the server partition to the client partition; and
>
> responsive to the client partition accepting the identifier, mapping the logical resource into a logical address space of the client partition, wherein the mapping is performed by the client partition.

Regarding claim 1, the Examiner states the following:

> As per claim 1, *Armstrong* et al. teach
>
> a method for managing shared resources in a logical partitioned data processing system - fig. 1: data processing apparatus; fig. 2: logical partitions with partition manager for shared services; col. 1, lines 43-67; col. 4, line 55 to col. 5, line 65.
>
> granting, by a server partition in the logical partitioned data processing system, a logical resource owned by the server partition to a client partition in the logical partitioned data processing system - fig. 2, the primary partition (A) is the server partition, the secondary partition (B)/(C) is the client partition; col. 5, lines 23-65. communicating an identifier from the server partition to the client partition; and responsive to the client partition accepting the identifier, mapping, the logical resource into a logical address space of the client partition, wherein the mapping is performed by the client partition - col. 1, line 52-67 (a partition manager or hypervisor manages the logical partitions and facilitates the allocation of resources to different logical partitions .... maintains separate virtual memory address spaces for the various logical partitions so that the memory utilized by each logical partitions is fully independent of the other logical partitions. One or more address translation tables are typically used

> by a partition manager to map addresses from each virtual address space to different addresses in the physical, or real, address space of the computer...so that the shared memory can be accessed directly by the logical partition. Examiner interprets that the Address Translation Tables store IDs that the clients can use to access resources across partitions); col. 5, lines 23-65 (each logical partition 40-44 executes in a separate memory space, represented by virtual memory 60. Moreover each logical partition is statically and/or dynamically allocated a portion of the available resources in computer 10... Resources can be allocated in a number of manners... ); col. 7, line 66 to col. 8, line 36.

A prior art reference anticipates the claimed invention under 35 U.S.C. § 102 only if every element of a claimed invention is identically shown in that single reference, arranged as they are in the claims. *In re Bond*, 910 F.2d 831, 832, 15 U.S.P.Q.2d 1566, 1567 (Fed. Cir. 1990). All limitations of the claimed invention must be considered when determining patentability. *In re Lowry*, 32 F.3d 1579, 1582, 32 U.S.P.Q.2d 1031, 1034 (Fed. Cir. 1994). Anticipation focuses on whether a claim reads on the product or process a prior art reference discloses, not on what the reference broadly teaches. *Kalman v. Kimberly-Clark Corp.*, 713 F.2d 760, 218 U.S.P.Q. 781 (Fed. Cir. 1983). In this case, each and every feature of the presently claimed invention is not identically shown in the cited reference, arranged as they are in the claims.

*Armstrong* is directed to a multiprocessor environment where one processor can initiate the resetting of another processor. When a target processor "hangs" or encounters a partial failure, a service processor sets a flag at a static memory location in a local storage for the target processor. The service processor then invalidates each entry in an address translation table allocated to the target processor. Once each entry in the address table has been invalidated, the target processor regains control of its resources, and enters its known initial state. According to *Armstrong*:

> To initiate a processor reset in the manner described herein, a first processor (referred to herein as a "source processor"), which desires to initiate a processor reset of another processor (referred to herein as a "target processor"), typically must generate a reset request and a memory access interrupt for the target processor. In FIG. 2, an exemplary source processor allocated to primary logical partition 40 is illustrated at "A", and an exemplary target processor allocated to secondary logical partition 42 is illustrated at "B". However, it should be appreciated that source and target processors may be allocated to the same logical partition in some implementations.

> In the illustrated embodiment, generation of a reset request is implemented via setting a flag located at a static memory location in the local storage for the target processor. The reset flag may alternatively be implemented in any other memory storage device that is accessible (at least indirectly) to both the source and target processors. Moreover, other manners of generating a reset request or otherwise indicating to a target processor that a reset is requested will be appreciated by one of ordinary skill in the art having the benefit of the instant disclosure.
>
> Also in the illustrated embodiment, generation of a memory access interrupt is implemented by invalidating, with the source processor, every entry in an address translation table associated with the target processor. Doing so ensures that the next time the target processor attempts to access any memory address (be it to retrieve a next instruction or to access data stored in memory), a memory access interrupt will be generated. In addition, to maintain coherency, it is desirable to update any caching mechanisms (such as TLB's) to invalidate any cached entries from an invalidated address translation table, and/or to update any other caching mechanisms that cache data and/or instructions associated with any such invalidated entries.

Col. 7, ll. 14-48

> FIG. 3 illustrates in greater detail a suitable implementation of address translation table 92 allocated to logical partition 42 and used by target processor B (FIG. 2). As is well known in the art, an address translation table includes a plurality of entries, e.g., entry 102, including a plurality of fields 104, 106, 108 and 110.

Col. 7, l. 66-Col. 8, l. 4

> Each entry 102 further includes a valid field 108 storing a bit that indicates whether or not the entry represents a valid mapping of a virtual page to a real page. It is this bit that is cleared by a source processor whenever it is desired to generate a memory access interrupt on a target processor that utilizes address translation table 92.

Col. 8, ll. 22-27

**A.1.a.** ***Armstrong*** **does not disclose the claim 1 feature of "*granting, by a server partition in the logical partitioned data processing system*, a logical resource owned by the server partition to a client partition in the logical partitioned data processing system"**

Contrary to the Examiner's assertions, *Armstrong* does not anticipate claim 1 because *Armstrong* does not disclose the claim 1 feature of "*granting*, *by a server partition* in the logical

partitioned data processing system, a logical resource owned by the server partition to a client partition in the logical partitioned data processing system." With regard to this claim feature, the Examiner cites the following section from *Armstrong*:

> Each logical partition 40-44 executes in a separate memory space, represented by virtual memory 60. Moreover, each logical partition 40-44 is statically and/or dynamically allocated a portion of the available resources in computer 10. For example, each logical partition is allocated one or more processors 12, as well as a portion of the available memory space for use in virtual memory 60. Logical partitions can share specific hardware resources such as processors, such that a given processor is utilized by more than one logical partition. In the alternative hardware resources can be allocated to only one logical partition at a time.
>
> Additional resources, e.g., mass storage, backup storage, user input, network connections, and the like, are typically allocated to one or more logical partitions in a manner well known in the art. Resources can be allocated in a number of manners, e.g., on a bus-by-bus basis, or on a resource-by-resource basis, with multiple logical partitions sharing resources on the same bus. Some resources may even be allocated to multiple logical partitions at a time. FIG. 2 illustrates, for example, three logical buses 62, 64 and 66, with a plurality of resources on bus 62, including a direct access storage device (DASD) 68, a control panel 70, a tape drive 72 and an optical disk drive 74, allocated to primary logical partition 40. Bus 64, on the other hand, may have resources allocated on a resource-by-resource basis, e.g., with local area network (LAN) adaptor 76, optical disk drive 78 and DASD 80 allocated to secondary logical partition 42, and LAN adaptors 82 and 84 allocated to secondary logical partition 44. Bus 66 may represent, for example, a bus allocated specifically to logical partition 44, such that all resources on the bus, e.g., DASD's 86 and 88, are allocated to the same logical partition.
>
> It will be appreciated that the illustration of specific resources in FIG. 2 is merely exemplary in nature, and that any combination and arrangement of resources may be allocated to any logical partition in the alternative. Moreover, it will be appreciated that in some implementations resources can be reallocated on a dynamic basis to service the needs of other logical partitions. Furthermore, it will be appreciated that resources may also be represented in terms of the input/output processors (IOP's) used to interface the computer with the specific hardware devices.

*Armstrong*, Col. 5, ll. 23-65

As can be seen, the cited section of *Armstrong* discloses a known method for allocating resources within a logically partitioned data processing system. The cited paragraphs describe the

traditional function of the partition manager, or hypervisor. *Armstrong* describes the hypervisor as follows:

> A shared resource, often referred to as a "hypervisor" or partition manager, manages the logical partitions and facilitates the allocation of resources to different logical partitions. As a component of this function, a partition manager maintains separate virtual memory address spaces for the various logical partitions so that the memory utilized by each logical partition is fully independent of the other logical partitions. One or more address translation tables are typically used by a partition manager to map addresses from each virtual address space to different addresses in the physical, or real, address space of the computer. Then, whenever a logical partition attempts to access a particular virtual address, the partition manager translates the virtual address to a real address so that the shared memory can be accessed directly by the logical partition.

*Armstrong*, Col. 1, ll 52-67

A partition manager is not a *server partition* in the logical partitioned data processing system. This distinction is clear from *Armstrong*'s Figure 2, the supporting text, and *Armstrong*'s description of the hypervisor/partition manager. According to *Armstrong*'s Figure 2, the partition manager is indicated by reference number 46. The partitions on the other hand are indicated by reference number 40-44. Primary partition 40 includes only enough partition management functionality to initiate processor resets of the other partitions in the event of a processor hang. Primary partition 40 does not "grant" or allocate anything. Any granting of resources is done by partition manager 46.

Contrary to both the teaching of *Armstrong*, and the Appellants' own disclosure, the Examiner maintains that allocation of resources by *Armstrong*'s partition manager reads on claim 1's "*granting, by a server partition* in the logical partitioned data processing system, a logical resource owned by the server partition." This is simply not true. As shown, a partition manager is not a server partition. While the partition manager is initially responsible for allocation of resources between partitions, *Armstrong* stops here. Claim 1 takes this one step further. Once a resource is allocated to a partition, that server partition is then allowed to share the resource with a client partition.

A partition manager is not a partition, based on either of the Appellants' definition or that of *Armstrong*. *Armstrong* teaches that only the partition manager grants resources. Because *Armstrong* does not disclose the claim 1 feature of "*granting, by a server partition* in the logical

partitioned data processing system, a logical resource owned by the server partition to a client partition in the logical partitioned data processing system," *Armstrong* does not anticipate claim 1 under 35 USC 102(b).

**A.1.b. *Armstrong* does not disclose the claim 1 feature of "*granting*, by a server partition in the logical partitioned data processing system, a logical resource owned *by the server partition to a client partition* in the logical partitioned data processing system"**

Contrary to the Examiner's assertions, *Armstrong* does not anticipate claim 1 because *Armstrong* does not disclose the claim 1 feature of "*granting*, by a server partition in the logical partitioned data processing system, a logical resource owned *by the server partition to a client partition* in the logical partitioned data processing system." Again, the Examiner relies on *Armstrong*'s Col. 5, ll. 23-65 in attempt to anticipate the claim feature.

As shown above, *Armstrong*'s partition manager is not a partition. Furthermore, as shown above, *Armstrong*'s primary partition possesses only enough functionality of the partition manager to allow the primary partition to reset other partitions in the event of a processor hang. Prior to allocation by the primary partition, the resource is not "owned" by any partition. Therefore, *Armstrong* cannot teach that granting *by the server partition to a client partition.*

Because *Armstrong* does not disclose the claim 1 feature of "*granting*, by a server partition in the logical partitioned data processing system, a logical resource owned *by the server partition to a client partition* in the logical partitioned data processing system," *Armstrong* does not anticipate claim 1 under 35 USC 102(b).

**A.1.c. *Armstrong* does not disclose the claim 1 feature of "*granting*, by a server partition in the logical partitioned data processing system, *a logical resource owned by the server partition* to a client partition in the logical partitioned data processing system"**

Contrary to the Examiner's assertions, *Armstrong* does not anticipate claim 1 because *Armstrong* does not disclose the claim 1 feature of "*granting*, by a server partition in the logical partitioned data processing system, *a logical resource owned by the server partition* to a client partition in the logical partitioned data processing system." Again, the Examiner relies on *Armstrong*'s Col. 5, ll. 23-65 in attempt to anticipate the claim feature.

According to *Armstrong*, the partition manager allocates resources to partitions. The partitions then maintain exclusive control over any allocated resources. *Armstrong* does not teach that resources allocated to one partition are granted to another partition.

The only access that *Armstrong*'s primary partition has to the secondary partitions are for generating a reset request via setting a flag located at a static memory and invalidating address translation table entries. This processor reset, however, is not a "granting of a resource." The primary processor does not have use of the address translation table. The primary partition does not have access to the memory affected by the flag setting. Rather, the primary partition is *granted limited access* in order to reset a processor associated with the underlying resources. This granting of a limited access to a resource is not a "granting of a resource." Because *Armstrong* does not disclose the claim 1 feature of "*granting*, by a server partition in the logical partitioned data processing system, *a logical resource owned by the server partition to a client partition* in the logical partitioned data processing system," *Armstrong* does not anticipate claim 1 under 35 USC 102(b).

Because *Armstrong* does not disclose the claim 1 feature of "*granting*, by a server partition in the logical partitioned data processing system, *a logical resource owned by the server partition* to a client partition in the logical partitioned data processing system," *Armstrong* does not anticipate claim 1 under 35 USC 102(b).

**A.1.d. *Armstrong* does not disclose the claim 1 feature of "mapping the logical resource into a logical address space of the client partition, wherein the mapping is performed by the client partition."**

Contrary to the Examiner's assertions, *Armstrong* does not anticipate claim 1 because *Armstrong* does not disclose the claim 1 feature of "mapping the logical resource into a logical address space of the client partition, wherein the mapping is performed by the client partition." With regard to this claim feature, the Examiner cites the following section from *Armstrong*:

> A shared resource, often referred to as a "hypervisor" or partition manager, manages the logical partitions and facilitates the allocation of resources to different logical partitions. As a component of this function, a partition manager maintains separate virtual memory address spaces for the various logical partitions so that the memory utilized by each logical partition is fully independent of the other logical partitions. One or more address translation tables are typically used by a partition manager to map

> addresses from each virtual address space to different addresses in the physical, or real, address space of the computer. Then, whenever a logical partition attempts to access a particular virtual address, the partition manager translates the virtual address to a real address so that the shared memory can be accessed directly by the logical partition.

*Armstrong*, Col. 1, ll. 52-67

> FIG. 3 illustrates in greater detail a suitable implementation of address translation table 92 allocated to logical partition 42 and used by target processor B (FIG. 2). As is well known in the art, an address translation table includes a plurality of entries, e.g., entry 102, including a plurality of fields 104, 106, 108 and 110. Address translation in the illustrated embodiment occurs on a page-by-page basis, e.g., with a page size of 4096 bytes. Each entry 102 thus matches a page of virtual memory address to a corresponding page of real memory addresses in the memory system. The "page" of a memory address is typically identified by those bits from the memory address other than the lowest order number of bits corresponding to the page size. Thus, for a page size of 4096, as well as a 64-bit memory address space, a page is identified by the upper 42 bits (bits 0-41, where bit 0 is the MSB), with the low order 12 bits (bits 42-63) utilized to specify a particular memory address in an identified page. As such, in the illustrated implementation, field 104 of each entry 102 includes a 42-bit virtual page number, with entry 106 including a 42-bit real page number to which the virtual page is mapped. It should be appreciated that either or both of the virtual and real memory address spaces may have differing sizes consistent with the invention.
>
> Each entry 102 further includes a valid field 108 storing a bit that indicates whether or not the entry represents a valid mapping of a virtual page to a real page. It is this bit that is cleared by a source processor whenever it is desired to generate a memory access interrupt on a target processor that utilizes address translation table 92.
>
> Additional information, represented by field 110, may also be stored within an entry 102 in an address translation table 92. Typically, such additional information includes various protection bits, as well as reference, change, address compare and/or other information known in the art. It should be appreciated that other data structures may be utilized in an address translation scheme consistent with the invention.

*Armstrong*, Col. 7, l. 66-col. 8, l. 36

As can be seen, the cited section of *Armstrong* discloses a known method for allocating resources within a logically partitioned data processing system. The cited paragraphs describe the traditional function of the partition manager, or hypervisor.

A partition manager is not a *server partition* in the logical partitioned data processing system. This distinction is clear from *Armstrong*'s Figure 2, the supporting text, and *Armstrong*'s description of the hypervisor/partition manager. According to *Armstrong*'s Figure 2, the partition manager is indicated by reference number 46. The partitions on the other hand are indicated by reference number 40-44. Primary partition 40 includes only enough partition management functionality to initiate processor resets of the other partitions in the event of a processor hang. Primary partition 40 does not "grant" or allocate anything. Any granting of resources is done by partition manager 46.

Contrary to both the teaching of *Armstrong*, and the Appellants' own disclosure, the Examiner maintains that mapping of resources by *Armstrong*'s partition manager reads on claim 1's "mapping the logical resource into a logical address space of the client partition, wherein the mapping is performed by the client partition." This is simply not true. As shown, a partition manager is not a server partition. While the partition manager is initially responsible for allocation of resources between partitions, *Armstrong* stops here. Claim 1 takes this one step further. Once a resource is allocated to a partition, the server partition is then allowed to share the resource with a client partition. The server partition then maps "the logical resource into a logical address space of the client partition."

A partition manager is not a partition, based on either of the Appellants' definition or that of *Armstrong*. *Armstrong* teaches that only the partition manager grants resources. Because *Armstrong* does not disclose the claim 1 feature of "mapping the logical resource into a logical address space of the client partition, wherein the mapping is performed by the client partition," *Armstrong* does not anticipate claim 1 under 35 USC 102(b).

### A.2. Claim 2

Claim 2 is representative of this claim grouping. Claim 2 is as follows:

> 2. The method of claim 1, further comprising:
>
> generating the identifier for the logical resource, wherein the identifier is generated by a hypervisor.

Claim 2 introduces the claim feature of a hypervisor. *Armstrong* discloses a hypervisor as partition manager 46. The Examiner has previously equated the hypervisor to one of the Appellants' partitions, a position that is against both the Appellants' claims and *Armstrong*'s

teachings. However, assuming for argument's sake that the Examiner is correct, the introduction of a hypervisor in claim 2 leaves the Examiner's rejection without a corresponding feature in the *Armstrong* reference.

That is, if *Armstrong*'s partition manager is incorrectly assumed to be a partition, then *Armstrong* has no corresponding structure that can be equated to Appellants' hypervisor, as recited in claim 2. Therefore, the Examiner would have failed to point to "every element of a claimed invention…identically shown in that single reference." The Examiner would therefore have failed to recite a *prima facie* case of anticipation under 35 U.S.C. 102(b).

Conversely, if *Armstrong*'s partition manager is correctly equated with claim 2's hypervisor, then *Armstrong*'s partition manager cannot also be a partition for the purposes of underlying dependent claim 1. This leads to the same conclusion: the Examiner would have failed to point to "every element of a claimed invention is identically shown in that single reference." The Examiner would therefore have failed to recite a *prima facie* case of anticipation under 35 U.S.C. 102(b).

### A.3. Claim 5

Claim 5 is representative of this claim grouping. Claim 5 is as follows:

> 5. The method of claim 1, further comprising:
> returning, by the client partition, the logical resource to the server partition.

As shown above, *Armstrong*'s partition manager is not a *server partition* in the logical partitioned data processing system. This distinction is clear from *Armstrong*'s Figure 2, the supporting text, and *Armstrong*'s description of the hypervisor/partition manager. According to *Armstrong*'s Figure 2, the partition manager is indicated by reference number 46. The partitions on the other hand are indicated by reference number 40-44. Primary partition 40 includes only enough partition management functionality to initiate processor resets of the other partitions in the event of a processor hang.

Thus, in the event that a partition is to release resources, *Armstrong* teaches that those resources would be returned to the partition manager, not the server partition. Because *Armstrong* does not disclose the claim 5 feature of "returning, by the client partition, the logical resource to the server partition," *Armstrong* does not anticipate claim 5 under 35 USC 102(b).

### A.4. Claim 6

Claim 6 is representative of this claim grouping. Claim 6 is as follows:

> 6. The method of claim 5, further comprising:
> rescinding, by the server partition, the logical resource.

As shown above, *Armstrong*'s partition manager is not a *server partition* in the logical partitioned data processing system. This distinction is clear from *Armstrong*'s Figure 2, the supporting text, and *Armstrong*'s description of the hypervisor/partition manager. According to *Armstrong*'s Figure 2, the partition manager is indicated by reference number 46. The partitions on the other hand are indicated by reference number 40-44. Primary partition 40 includes only enough partition management functionality to initiate processor resets of the other partitions in the event of a processor hang.

Thus, in the event that a partition is to release resources, *Armstrong* teaches that those resources would be returned to the partition manager, not the server partition. Because *Armstrong* does not disclose the claim 6 feature of "rescinding, by the server partition, the logical resource," *Armstrong* does not anticipate claim 5 under 35 USC 102(b).

### A.5. Claim 8

Claim 8 is representative of this claim grouping. Claim 8 is as follows:

> 8. The method of claim 7, further comprising:
> preventing translation tables in the client partition from containing references to a physical address of the logical resource.

With regard to this claim feature, the Examiner cites the following section from *Armstrong*:

> Detection of a reset request is typically implemented within an interrupt handler that is executed by a target processor in response to a memory access interrupt. As a result, for those situations in which a memory access interrupt is generated for a reason other than to initiate a reset of the target processor, the target processor can handle the interrupt in an appropriate manner, and often with little additional overhead associated with determining whether a reset operation should be performed as a result of the interrupt.
>
> A memory access interrupt may be considered to include any type of interrupt that is generated responsive to a memory access attempt by the target processor. Particularly given the general necessity for a processor to

> always be capable of accessing memory, a memory access interrupt is often further characterized as being incapable of being disabled during the operation of the target processor. As a consequence, unlike external interrupts such as IPI's and the like which are capable of being disabled in some instances, a reset operation can be initiated on a target processor via a memory access interrupt irrespective of whether other interrupts are disabled on the processor.
>
> While other alternative memory access interrupt implementations may also be utilized consistent with the invention, one particularly useful implementation relies on a type of memory access interrupt that is generated in response to an attempt by a target processor to access a virtual memory address in a virtual memory address space that is not mapped by any entry in an address translation table. Generation of a memory access interrupt then typically requires only that one or more entries in the address translation table be invalidated to ensure that a subsequent access to the virtual memory address space will attempt to access an unmapped virtual memory address.
>
> Therefore, consistent with one aspect of the invention, a processor may be reset by generating a reset request for the processor, generating a memory access interrupt on the processor, and resetting the processor during handling of the memory access interrupt by the processor responsive to detection of the reset request.
>
> These and other advantages and features, which characterize the invention, are set forth in the claims annexed hereto and forming a further part hereof. However, for a better understanding of the invention, and of the advantages and objectives attained through its use, reference should be made to the Drawings, and to the accompanying descriptive matter, in which there is described exemplary embodiments of the invention.

*Armstrong*, Col. 3, ll. 9-56.

This section of *Armstrong* states only what the Appellants have previously pointed out: namely that when a target processor "hangs" or encounters a partial failure, *Armstrong*'s service processor sets a flag at a static memory location in a local storage for the target processor. The service processor then invalidates each entry in an address translation table allocated to the target processor. Nothing in this cited section, or elsewhere teaches translation tables in the client partition that are prevented from containing references to a physical address of the logical resource

In direct contradiction of the Examiner's statement, translation tables in *Armstrong* unequivocally contain a mapping of logical addresses to physical addresses. Figure 3 is an address translation table that can be one of address translation tables 90-94 of Figure 2. Each of these

address translation tables corresponds to one of partitions 40-44. Figure 3 includes real page number 106. Real page number 106 is described as follows:

> Address translation in the illustrated embodiment occurs on a page-by-page basis, e.g., with a page size of 4096 bytes. Each entry 102 thus matches a page of virtual memory address to a corresponding page of real memory addresses in the memory system. The "page" of a memory address is typically identified by those bits from the memory address other than the lowest order number of bits corresponding to the page size. Thus, for a page size of 4096, as well as a 64-bit memory address space, a page is identified by the upper 42 bits (bits 0-41, where bit 0 is the MSB), with the low order 12 bits (bits 42-63) utilized to specify a particular memory address in an identified page. As such, in the illustrated implementation, field 104 of each entry 102 includes a 42-bit virtual page number, with entry 106 including a 42-bit real page number to which the virtual page is mapped. It should be appreciated that either or both of the virtual and real memory address spaces may have differing sizes consistent with the invention.

*Armstrong*, col. 8, ll. 5-22.

Each of partitions 40-44 is associated with an address resolution table similar to the one of Figure 3. Therefore, each of the partitions 40-44 has access to the real addresses and virtual addresses for the assigned logical resources. Any assertion otherwise is erroneous. Because *Armstrong* does not disclose the claim 8 feature of "preventing translation tables in the client partition from containing references to a physical address of the logical resource," *Armstrong* does not anticipate claim 8 under 35 USC 102(b).

### A.6. Claim 22

The Examiner rejects claim 22 under 25 USC 102(a). Claim 22 recites features in addition to those found in claim 1. However, the Examiner makes no statements regarding the features of claim 22 or any corollaries in *Armstrong*.

## B. GROUND OF REJECTION 2 (Claim 22)

The Examiner rejected claim 22 under 35 U.S.C. §102(b) as being anticipated by *Armstrong*. Appellants request that the Board of Patent Appeals and Interferences overturn this clearly erroneous rejection.

Claim 22 is representative of this claim grouping. Claim 22 is as follows:

> 22. The method of claim 1, wherein the client partition is a first client

> partition, the method further comprising:
>
> communicating an identifier from the server partition to the first client partition, wherein the identifier is a cookie that identifies the logical resource owned by the server partition;
>
> responsive to the first client partition accepting the identifier, mapping the logical resource into a logical address space of the first client partition, wherein the mapping is performed by the first client partition;
>
> granting, by the first client partition in the logical partitioned data processing system, the logical resource owned by the server partition to a second client partition in the logical partitioned data processing system;
>
> communicating an identifier from the first client partition to the second client partition; and
>
> responsive to the second client partition accepting the identifier, mapping the logical resource into a logical address space of the second client partition, wherein the mapping is performed by the second client partition.

The Examiner bears the burden of establishing a *prima facie* case of obviousness based on prior art when rejecting claims under 35 U.S.C. § 103. *In re Fritch*, 972 F.2d 1260, 23 U.S.P.Q.2d 1780 (Fed. Cir. 1992). The prior art reference (or references when combined) must teach or suggest all the claim limitations. *In re Royka,* 490 F.2d 981, 180 USPQ 580 (CCPA 1974). In determining obviousness, the scope and content of the prior art are... determined; differences between the prior art and the claims at issue are... ascertained; and the level of ordinary skill in the pertinent art resolved. Against this background the obviousness or non-obviousness of the subject matter is determined. *Graham v. John Deere Co.*, 383 U.S. 1 (1966). "Often, it will be necessary for a court to look to interrelated teachings of multiple patents; the effects of demands known to the design community or present in the marketplace; and the background knowledge possessed by a person having ordinary skill in the art, all in order to determine whether there was an apparent reason to combine the known elements in the fashion claimed by the patent at issue." *KSR Int'l. Co. v. Teleflex*, Inc., No. 04-1350 (U.S. Apr. 30, 2007). "Rejections on obviousness grounds cannot be sustained by mere conclusory statements; instead, there must be some articulated reasoning with some rational underpinning to support the legal conclusion of obviousness." *Id.* (citing *In re Kahn*, 441 F.3d 977, 988 (CA Fed. 2006)).

Regarding claim 22, the Examiner states the following:

> As per claim 22, Amstrong et al. teach
> communicating an identifier from the server partition to the client partition; and responsive to the client partition accepting the identifier, mapping, the logical resource into a logical address space of the client partition, wherein the mapping is performed by the client partition - col. 1,

> line 52-67 (a partition manager or hypervisor manages the logical partitions and facilitates the allocation of resources to different logical partitions .... maintains separate virtual memory address spaces for the various logical partitions so that the memory utilized by each logical partitions is fully independent of the other logical partitions. One or more address translation tables are typically used by a partition manager to map addresses from each virtual address space to different addresses in the physical, or real, address space of the computer...so that the shared memory can be accessed directly by the logical partition. Examiner interprets that the Address Translation Tables store IDs that the clients can use to access resources across partitions); col. 5, lines 23-65 (each logical partition 40-44 executes in a separate memory space, represented by virtual memory 60. Moreover each logical partition is statically and/or dynamically allocated a portion of the available resources in computer 10... Resources can be allocated in a number of manners... ); col. 7, line 66 to col. 8, line 36 (...the address translation table 92 with items virtual page number, real page number...The virtual page number can be interpreted as an identifier).
>
> However, Amstrong et al. do not teach the identifier is a cookie. *Farber* discloses the sharing of resources and the partitioned cache into separate areas for each subscriber - pars. 336-338; The resources relies on a so-called cookie - pars. 85, 329. Thus, it would have been obvious to one of ordinary skill in the art at the time of the invention to combine Amstrong's teaching with *Farber*'s teaching in order to allow different ways of identifying of resources that partitions can request to share.

**B.1. The combination of references does not disclose the claim 22 feature of "communicating an identifier from the first client partition to the second client partition"**

The Appellants point out the heretofore unacknowledged antecedent basis issue with the current claim limitation. Appellants will correct this defect upon a favorable decision from the board. Appellants instead ask the Board to consider the merits of claim 22, regardless of the to-be-cured defect.

Claim 22 recites additional limitations to make clear that identifiers and resources are granted from the server partition to the client partition, and not exclusively from the partition manager to one of the partitions. The claim 22 feature of "communicating an identifier from the first client partition to the second client partition" further delineates this grant of resources. The Examiner makes no statements regarding this feature of claim 22 or any corollaries in *Armstrong*. Instead, the Examiner's rejection recites only those features previously addressed in the underlying claim 1.

As shown above, *Armstrong*'s partition manager is not a *server partition* in the logical partitioned data processing system. This distinction is clear from *Armstrong*'s Figure 2, the supporting text, and *Armstrong*'s description of the hypervisor/partition manager. According to *Armstrong*'s Figure 2, the partition manager is indicated by reference number 46. The partitions on the other hand are indicated by reference number 40-44. Primary partition 40 includes only enough partition management functionality to initiate processor resets of the other partitions in the event of a processor hang.

*Farber* does not overcome the deficiencies of *Armstrong*. *Farber* discloses a way for servers in a computer network to off-load their processing of requests for selected resources by determining a different server (a "repeater") to process those requests. The selection of the repeater can be made dynamically, based on information about possible repeaters. Resource requests made by clients of origin servers in a network are intercepted by reflector mechanisms and selectively reflected to other servers called repeaters. The reflectors select a best repeater from a set of possible repeaters and redirect the client to the selected best repeater. The client then makes the request of the selected best repeater. The resource is possibly rewritten to replace at least some of the resource identifiers contained therein with modified resource identifiers designating the repeater instead of the origin server.

Claim 22 introduces the claim feature of a second client partition. The first client partition grants the resource to the second client partition, after the initial grant of the resource to the first partition. The combination of references does not disclose this feature of claim 1. Indeed, the Examiner makes no attempt to correlate this claim 1 feature to any disclosure within the cited references. Accordingly, under the standards of *KSR Int'l.*, the Examiner failed to state a *prima facie* obviousness rejection against claim 22.

**B.2. The combination of references does not disclose the claim 22 feature of "mapping the logical resource into a logical address space of the second client partition, wherein the mapping is performed by the second client partition"**

Claim 22 recites additional limitations to make clear that identifiers and resources are granted from the server partition to the client partition, and not exclusively from the partition manager to one of the partitions. The claim 22 feature of "mapping the logical resource into a logical address space of the second client partition, wherein the mapping is performed by the

second client partition" further delineates this grant of resources. The Examiner makes no statements regarding this feature of claim 22 or any corollaries in *Armstrong*. Instead, the Examiner's rejection recites only those features previously addressed in the underlying claim 1.

As shown above, *Armstrong*'s partition manager is not a *server partition* in the logical partitioned data processing system. This distinction is clear from *Armstrong*'s Figure 2, the supporting text, and *Armstrong*'s description of the hypervisor/partition manager. According to *Armstrong*'s Figure 2, the partition manager is indicated by reference number 46. The partitions on the other hand are indicated by reference number 40-44. Primary partition 40 includes only enough partition management functionality to initiate processor resets of the other partitions in the event of a processor hang.

*Farber* does not overcome the deficiencies of *Armstrong*. *Farber* discloses a way for servers in a computer network to off-load their processing of requests for selected resources by determining a different server (a "repeater") to process those requests. The selection of the repeater can be made dynamically, based on information about possible repeaters. Resource requests made by clients of origin servers in a network are intercepted by reflector mechanisms and selectively reflected to other servers called repeaters. The reflectors select a best repeater from a set of possible repeaters and redirect the client to the selected best repeater. The client then makes the request of the selected best repeater. The resource is possibly rewritten to replace at least some of the resource identifiers contained therein with modified resource identifiers designating the repeater instead of the origin server.

Claim 22 introduces the claim feature of a second client partition. The second client partition maps the logical resource into a logical address space of the second client partition, after the first client partition grants access of the resource to the second partition. The combination of references does not disclose this feature of claim 1. Indeed, the Examiner makes no attempt to correlate this claim 1 feature to any disclosure within the cited references. Accordingly, under the standards of *KSR Int'l.*, the Examiner failed to state a *prima facie* obviousness rejection against claim 22.

## C. CONCLUSION

As shown above, the Examiner has failed to state valid rejections against any of the claims. Therefore, Appellants request that the Board of Patent Appeals and Interferences reverse the rejections. Additionally, Appellants request that the Board direct the Examiner to allow the claims.

Date: February 18, 2009

Respectfully submitted,

/Brandon G. Williams/

Brandon G. Williams
Reg. No. 48,844
Yee & Associates, P.C.
P.O. Box 802333
Dallas, TX 75380
(972) 385-8777

## CLAIMS APPENDIX

The text of the claims involved in the appeal is as follows:

1. A method for managing shared resources in a logical partitioned data processing system, the method comprising:

granting, by a server partition in the logical partitioned data processing system, a logical resource owned by the server partition to a client partition in the logical partitioned data processing system;

communicating an identifier from the server partition to the client partition; and

responsive to the client partition accepting the identifier, mapping the logical resource into a logical address space of the client partition, wherein the mapping is performed by the client partition.

2. The method of claim 1, further comprising:

generating the identifier for the logical resource, wherein the identifier is generated by a hypervisor.

3. The method of claim 2, wherein the identifier is unique within the client partition.

4. The method of claim 2, wherein the identifier cannot be used to access the logical resource outside the client partition.

5. The method of claim 1, further comprising:

returning, by the client partition, the logical resource to the server partition.

6. The method of claim 5, further comprising:

rescinding, by the server partition, the logical resource.

7. The method of claim 1, further comprising:

responsive to a determination, at the server partition, that the client partition is incapable of gracefully returning the logical resource, performing a forced rescind operation.

8. The method of claim 7, further comprising:

preventing translation tables in the client partition from containing references to a physical address of the logical resource.

9. The method of claim 1, further comprising:

responsive to a failure of the server partition, notifying the client partition of the failure of the server partition;

recovering outstanding shared logical resources for the server partition; and

restarting the server partition.

10. The method of claim 9, further comprising:

delaying for a period of time prior to the step of recovering the outstanding shared logical resources for the server partition.

22. The method of claim 1, wherein the client partition is a first client partition, the method further comprising:

communicating an identifier from the server partition to the first client partition, wherein the identifier is a cookie that identifies the logical resource owned by the server partition;

responsive to the first client partition accepting the identifier, mapping the logical resource into a logical address space of the first client partition, wherein the mapping is performed by the first client partition;

granting, by the first client partition in the logical partitioned data processing system, the logical resource owned by the server partition to a second client partition in the logical partitioned data processing system;

communicating an identifier from the first client partition to the second client partition; and

responsive to the second client partition accepting the identifier, mapping the logical resource into a logical address space of the second client partition, wherein the mapping is performed by the second client partition.

## EVIDENCE APPENDIX

This appeal brief presents no additional evidence.

**RELATED PROCEEDINGS APPENDIX**

This appeal has no related proceedings.